

Web Service Integration Report

1. Introduction

The objective of this assignment is to integrate external web services into a backend application using secure and structured methods. The project focuses on consuming the Open Weather Map API to fetch current weather data for both single and multiple cities. The assignment aims to develop understanding of web APIs, service integration, authentication mechanisms, and secure API communication.

2. Implementation Steps

The major development and configuration steps completed in this assignment include: 1. Cloning the collaborative GitHub repository for team development. 2. Installing project dependencies using 'npm install'. 3. Creating and configuring an .env file containing sensitive API keys and server configuration values. 4. Implementing Express routes, controllers, and service layers for fetching weather data. 5. Integrating JWT authentication middleware to protect API endpoints. 6. Testing the application thoroughly using Postman to verify successful responses. 7. Handling errors, invalid inputs, and network failures using custom middleware and try/catch logic.

3. Tools and Technologies Used

- Node.js (server-side JavaScript runtime)
- Express.js (backend web framework)
- Axios (HTTP client for API consumption)
- JSON Web Tokens (JWT) for authentication and security
- Postman(API testing platform)
- GitHub (version control and collaboration)
- Visual Studio Code (development environment)

4. Results and Observations

- JWT token generation works correctly.
- Weather data retrieval for single and multiple cities was successful.
- Invalid cities produce error responses and are logged in logs/errors.log
- Postman tests confirmed authentication, headers, and API functionality.

4.1 Screenshot: Successful Single-City API Response

The screenshot displays the Postman interface with a workspace named 'API Network'. The left sidebar shows a collection of API endpoints under 'Weather API Integration - Group 9'. The selected endpoint is 'GET Weather - Single City'. The main panel shows the request details: a GET request to the URL '[[baseUri]]/api/weather/:city'. The 'Path Variables' table indicates that the variable 'city' has the value 'London'. The 'Body' tab is active, showing a JSON response with a status of '200 OK'. The response body is as follows:

```
{
  "success": true,
  "data": {
    "city": "London",
    "country": "GB",
    "coordinates": {
      "lat": 51.5085,
      "lon": -0.1257
    },
    "temperature": {
      "current": "3.73",
      "feels_like": "0.55",
      "min": "2.79",
      "max": "4.86",
      "unit": "celsius"
    }
  }
}
```

The bottom status bar shows the system time as 10:53 AM and the weather as 20°C Sunny.

4.2 Screenshot: Successful Multiple-City API Response

The screenshot displays the Postman interface with a workspace named 'API Network'. The left sidebar shows a collection of API endpoints under 'Weather API Integration - Group 9', with 'POST Get Weather - Multiple Cities' selected. The main panel shows the request details for this endpoint, which is a POST request to '{{\$baseUrl}}/api/weather/multiple'. The request body is a JSON array of city names: London, Paris, Tokyo, New York, and Sydney. The response is a 200 OK status with a JSON body indicating success and providing details for the first city, London.

Request Details:

- Method: POST
- URL: {{\$baseUrl}}/api/weather/multiple
- Body (raw):

```
1 {  
2   "cities": [  
3     "London",  
4     "Paris",  
5     "Tokyo",  
6     "New York",  
7     "Sydney"  
8   ]  
9 }
```

Response Details:

- Status: 200 OK
- Body (JSON):

```
1 {  
2   "success": true,  
3   "count": 5,  
4   "results": [  
5     {  
6       "city": "London",  
7       "success": true,  
8       "data": {  
9         "city": "London",  
10        "country": "GB",  
11      }  
12    }  
13   ]  
14 }
```

4.3 Screenshot: JWT Token Generation (Login)

The screenshot displays the Postman interface for a REST client. The left sidebar shows the 'My Workspace' with a collection named 'Weather API Integration - Group 9' expanded, revealing an 'Authentication' folder containing a 'Get Token' request. The main panel shows the details of this 'GET' request to '({baseUrl}) /api/auth/token'. The response is a 200 OK status with a response time of 23 ms and a body size of 432 B. The response body is a JSON object containing a success flag, a token, and an expiration time.

Request Details:

- Method: GET
- URL: ({baseUrl}) /api/auth/token

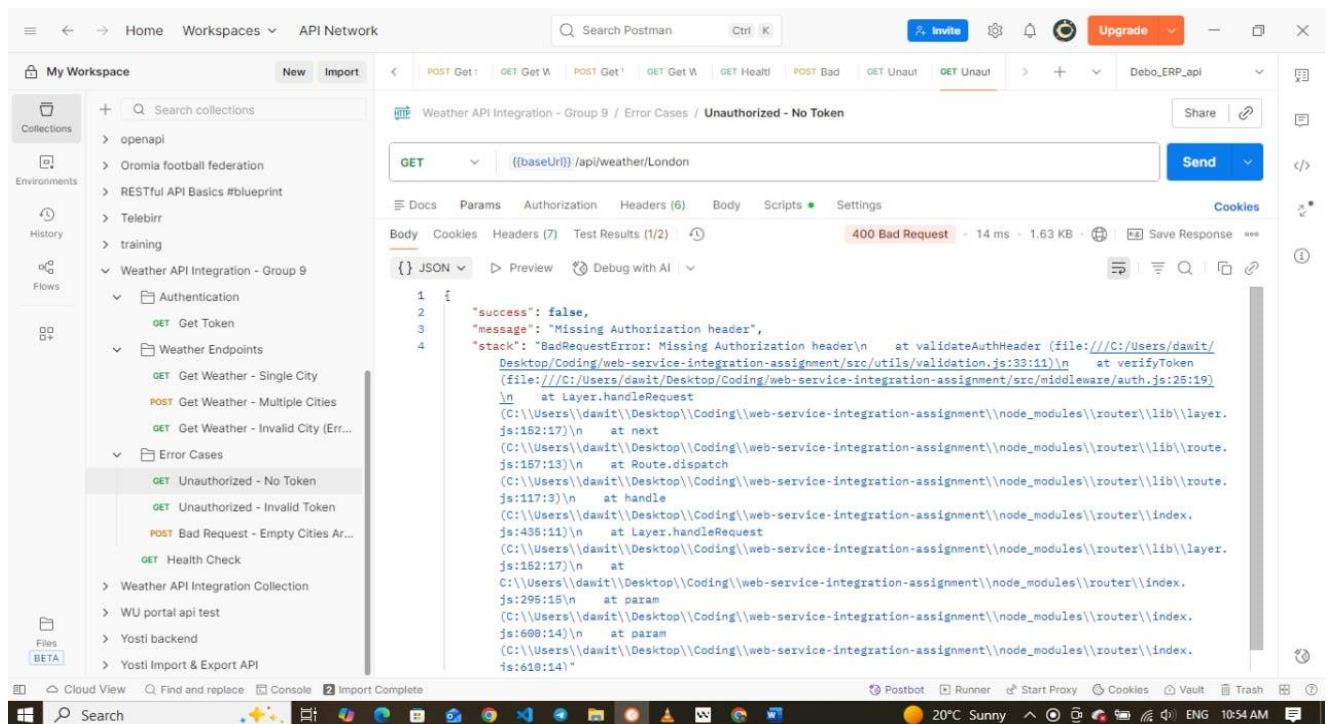
Response Details:

- Status: 200 OK
- Time: 23 ms
- Size: 432 B

Response Body (JSON):

```
1 {
2   "success": true,
3   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyY2xiIjoic3R1ZGVudCIsIm1hdCI6MTc2NDA1NzE0MiwiaXNjaXNzY0MDYwNzQyIiwiaWF0IjE6g-rdz_Cf_GLB1bgx2QW9xOn8Zj8hpRe9sFIes1Bw",
4   "expires_in": "1h"
5 }
```

4.4 Screenshot: Authorization Header (Bearer Token)



4.5 Screenshot: Empty Cities Array

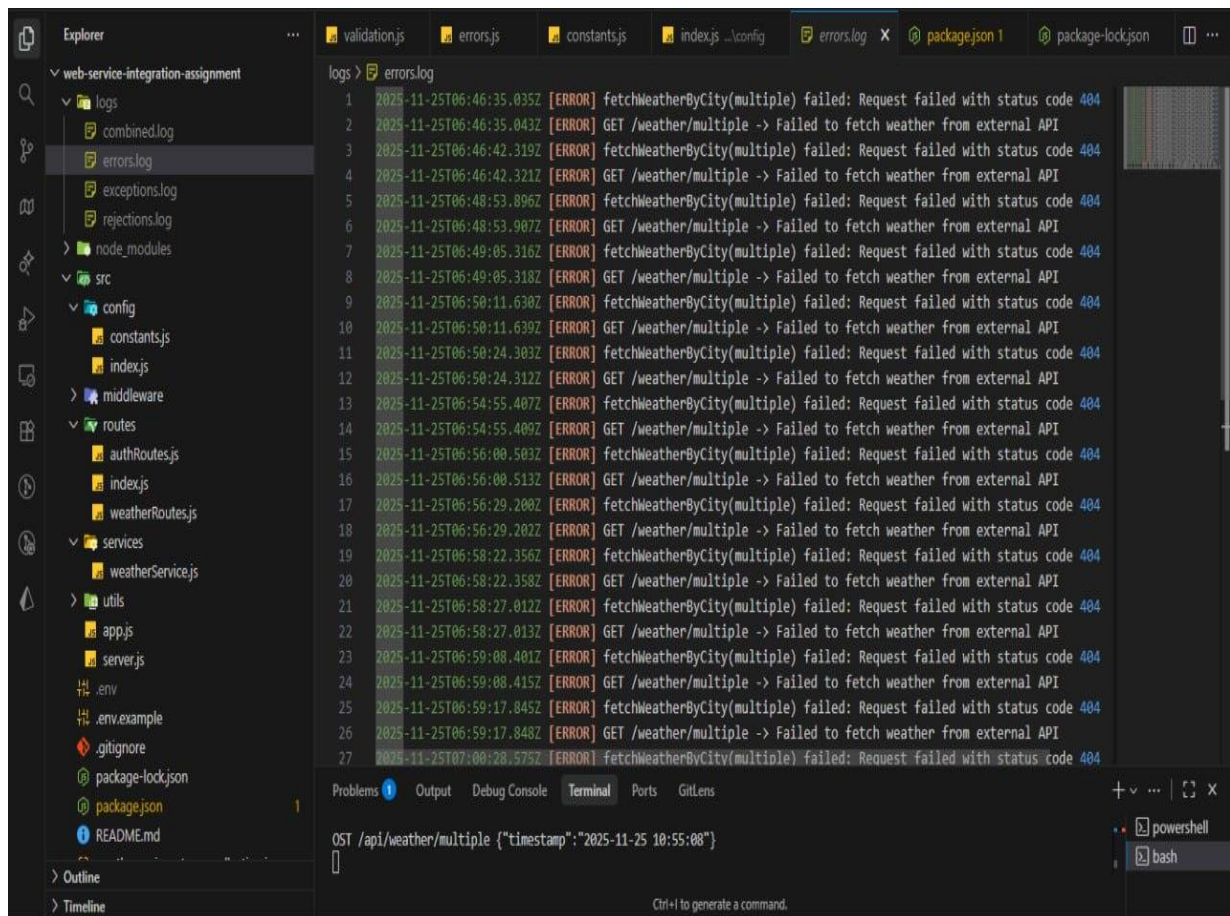
The screenshot shows the Postman application interface. On the left, the 'My Workspace' sidebar displays a collection of API requests under 'Weather API Integration - Group 9'. The 'Error Cases' folder is expanded, showing a 'POST Bad Request - Empty Cities Array' request. The main panel shows the details of this request, which is a POST to '({baseUrl})/api/weather/multiple'. The response is a 400 Bad Request, with a status of 29 ms and 1.66 KB. The response body is in JSON format, showing a 'success' field set to false and a 'message' field with the text: 'Please provide a 'cities' array in the request body.'. The 'stack' field contains a detailed error trace, starting from 'validateCitiesArray' in 'validation.js' and ending at 'index.js'. The Windows taskbar at the bottom shows the system time as 10:55 AM and the temperature as 20°C.

Postman Interface Screenshot:

- Workspace:** My Workspace
- API Network:** Weather API Integration - Group 9 / Error Cases / Bad Request - Empty Cities Array
- Request Method:** POST
- URL:** ({baseUrl})/api/weather/multiple
- Response Status:** 400 Bad Request
- Response Time:** 29 ms
- Response Size:** 1.66 KB
- Response Body (JSON):**

```
{  "success": false,  "message": "Please provide a 'cities' array in the request body.",  "stack": "BadRequestError: Please provide a 'cities' array in the request body.\n    at validateCitiesArray\n      (file:///C:/Users/dawit/Desktop/Coding/web-service-integration-assignment/src/utls/validation.js:10:11)\n    at file:///C:/Users/dawit/Desktop/Coding/web-service-integration-assignment/src/routes/weatherRoutes.js:41:5\n    at file:///C:/Users/dawit/Desktop/Coding/web-service-integration-assignment/src/utls/errors.js:62:19\n    at Layer.handleRequest\n      (C:/Users/dawit/Desktop/Coding/web-service-integration-assignment/node_modules/router/lib/layer.js:152:17)\n    at next\n      (C:/Users/dawit/Desktop/Coding/web-service-integration-assignment/node_modules/router/lib/route.js:157:13)\n    at verifyToken\n      (file:///C:/Users/dawit/Desktop/Coding/web-service-integration-assignment/src/middleware/auth.js:29:5)\n    at Layer.handleRequest\n      (C:/Users/dawit/Desktop/Coding/web-service-integration-assignment/node_modules/router/lib/layer.js:152:17)\n    at next\n      (C:/Users/dawit/Desktop/Coding/web-service-integration-assignment/node_modules/router/lib/route.js:157:13)\n    at Route.dispatch\n      (C:/Users/dawit/Desktop/Coding/web-service-integration-assignment/node_modules/router/lib/route.js:117:3)\n    at handle\n      (C:/Users/dawit/Desktop/Coding/web-service-integration-assignment/node_modules/router/index.js:435:11)"
```


4.6 Screenshot: Error Log



```
logs > errors.log
1 2025-11-25T06:46:35.035Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
2 2025-11-25T06:46:35.043Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
3 2025-11-25T06:46:42.319Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
4 2025-11-25T06:46:42.321Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
5 2025-11-25T06:48:53.096Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
6 2025-11-25T06:48:53.907Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
7 2025-11-25T06:49:05.316Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
8 2025-11-25T06:49:05.318Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
9 2025-11-25T06:50:11.630Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
10 2025-11-25T06:50:11.632Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
11 2025-11-25T06:50:24.303Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
12 2025-11-25T06:50:24.312Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
13 2025-11-25T06:54:55.407Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
14 2025-11-25T06:54:55.409Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
15 2025-11-25T06:56:00.503Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
16 2025-11-25T06:56:00.513Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
17 2025-11-25T06:56:29.200Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
18 2025-11-25T06:56:29.202Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
19 2025-11-25T06:58:22.356Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
20 2025-11-25T06:58:22.358Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
21 2025-11-25T06:58:27.012Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
22 2025-11-25T06:58:27.013Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
23 2025-11-25T06:59:08.401Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
24 2025-11-25T06:59:08.415Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
25 2025-11-25T06:59:17.845Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404
26 2025-11-25T06:59:17.848Z [ERROR] GET /weather/multiple -> Failed to fetch weather from external API
27 2025-11-25T07:00:28.575Z [ERROR] fetchWeatherByCity(multiple) failed: Request failed with status code 404

Problems 1 Output Debug Console Terminal Ports GitLens
OST /api/weather/multiple {\"timestamp\": \"2025-11-25 10:55:08\"}
Ctrl+I to generate a command.
```

5. Conclusion

This project successfully fulfills all requirements of the Web Services Integration assignment. It demonstrates secure API integration, proper error handling, data transformation, and logging

This assignment offered hands-on experience in integrating external web services into a backend system using Node.js and Express. Through the implementation of JWT authentication, secure API communication, and structured project architecture, we gained practical understanding of service integration and backend development. The collaborative workflow on GitHub also provided real-world exposure to version control, branch management, and team-based software development practices.